UNIVERSIDAD CÉSAR VALLEJO

FACULTAD DE INGENIERÍA Y ARQUITECTURA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

Algoritmo para la compresión de imágenes basado en algoritmo LZW y RLE

TESIS PARA OBTENER EL TÍTULO PROFESIONAL DE:

Ingeniero de Sistemas

AUTOR:

Borja Castañeda, Kevin Arnold (orcid.org/0000-0002-2479-8358)

ASESOR:

Dr. Chumpe Agesto, Juan Brues Lee (orcid.org/0000-0001-7466-9872)

LÍNEA DE INVESTIGACIÓN:

Sistemas de Información y Comunicaciones

LÍNEA DE RESPONSABILIDAD SOCIAL UNIVERSITARIA:

Desarrollo económico, empleo y emprendimiento

LIMA – PERÚ

2022

Dedicatoria

Dedicado a mis padres y docentes que me apoyaron a lo largo del desarrollo de esta tesis y a lo largo de mi carrera, para los que están y para los que se fueron.

Agradecimiento

Agradezco a mi asesor, Juan Chumpe y al docente Emigdio Alfaro que me guiaron a través de todo el camino para el desarrollo de este informe.

Índice de contenidos

Índice de tablas

Índice de anexos

Índice de abreviaturas

Resumen

Esta investigación contó con el siguiente problema de investigación: ¿Qué algoritmo presenta la mejor tasa de compresión, tiempo de compresión y descompresión en imágenes entre los algoritmos IBK, LZW y RLE?

El objetivo de esta investigación fue determinar qué algoritmo presenta la mejor tasa de compresión, tiempo de compresión y descompresión en imágenes entre los algoritmos IBK, LZW y RLE. Esta investigación tuvo un enfoque cuantitativo con un diseño experimental y un tipo de diseño preexperimental. Para lo cual se tomaron 150 imágenes en formato sin pérdida y se les aplicó cada uno de los algoritmos mencionados. Para medir los resultados se utilizó una ficha de recolección de datos en las que se llevó un registro del tamaño original de la imagen, el tamaño comprimido el tiempo de compresión y el tiempo de descompresión.

A partir de esto, se obtuvo que el algoritmo IBK obtuvo mejor tasa de compresión que los demás algoritmos, pero no se llegó a tener mejor tiempo de compresión ni un mejor tiempo de descompresión.

Palabras clave: compresión de imágenes, LZW, RLE, algoritmo.

Abstract

This research had the following research problem: Which algorithm presents the best compression rate, compression time and decompression in images among the IBK, LZW and RLE algorithms?

The objective of this research was to determine which algorithm presents the best compression rate, compression time and decompression in images among the IBK, LZW and RLE algorithms. This research had a quantitative approach with an experimental design and a pre-experimental design type. For which 150 images were taken in lossless format and each of the mentioned algorithms was applied to them. To measure the results, a data collection form was used to record the original size of the image, the compressed size, the compression time and the decompression time.

From this, it was obtained that the IBK algorithm obtained a better compression rate than the other algorithms, but it did not have a better compression time nor a better decompression time.

Keywords: Image compression, LZW, RLE, Algorithm,

I. INTRODUCCIÓN

En este capítulo se da a conocer la importancia de compresión de imágenes tanto como para su almacenaje o su transmisión a través de la red. De igual manera se proporcionan unos estudios previos que nos dan una idea sobre la realidad problemática. Finalmente se detallará el problema general y los problemas específicos con sus respectivas hipótesis para este proyecto de investigación.

Compartir imágenes se ha vuelto casi un hábito en nuestras actividades diarias, tanto como en el ámbito laboral como en el ámbito personal. Gupta, Bansal y Khanduja (2017) señalaron que el mundo se está inclinándose cada vez más hacia las redes sociales y la cantidad de datos que se comparten a través de internet está creciendo cada vez más día a día. Y que debido a que el ancho de banda de las redes es limitado es que se precisan de algoritmos de compresión eficientes que faciliten la transmisión de datos de manera rápida y eficiente.

Asimismo, Rahman, Hamada y Shin (2021) mencionaron que grandes cantidades de información son generadas a diario, debido a los avances en las telecomunicaciones y el problema del almacenaje en dispositivos y su transmisión por internet es un reto, por lo que la compresión de datos es esencial para el correcto manejo de la información. Y que a pesar de que hay muchas estrategias desarrolladas para lograr este fin, no hay un algoritmo en específico que trabaje bien con todos los tipos de datos.

A su vez, Birajdar Amit et al. (2019a) precisaron que las imágenes son unas de las representaciones de datos más populares y comunes. Siendo éstas usadas por profesionales y también de un uso personal que va desde las redes sociales hasta documentación oficial. Es así, que tanto como organizaciones o personas normales tienen la necesidad de almacenar gran cantidad de imágenes. Con los avances en tecnología para la captura de imágenes, el aumento en popularidad del contenido digital, la calidad y la resolución ha llevado a un aumento en el tamaño, siendo asi que la compresión de imágenes se ha vuelto una parte esencial para el procesamiento de estas. Y que a pesar de que hay muchas técnicas que se han desarrollado para este fin, no se puede hablar de que hay un mejor algoritmo ya que depende del tipo de imagen que se vaya a comprimir.

Del mismo modo, Alam et al. (2018) alegaron que como la web, móviles, escritorio y otras aplicaciones usan las imágenes para distintos propósitos, esto ha conllevado a que la compresión de imágenes se vuelva un tópico muy popular en

el análisis de imágenes y las ciencias de la computación. Y que, a pesar de que la compresión de imágenes es un concepto antiguo, aún sigue siendo un proceso que consume bastante tiempo y que ha abierto un nuevo campo de investigación en la compresión de imágenes.

Es de esta manera que la masificación de las imágenes en medios digitales y su proliferación a través de la red ha generado bastantes inconvenientes. Debido a esto, Mbewe Phyela y D. Asare Sampson (2017) afirmaron que unos de los problemas del gran tamaño de las imágenes digitales es lo dificultosos para transferir por la red, ya que imágenes grandes en las páginas web toman más tiempo de carga y le generan a los usuarios más costos en la transferencia de datos ya que sale más barato usar imágenes pequeñas y comprimidas que unas grandes y sin comprimir. Siendo asi que algunos tipos de archivos se comprimen mejor que otros, por ejemplo, que ciertos archivos de texto e imágenes en formato BMP llegan a comprimirse en un 90%.

Dejando de lado el ámbito personal, Rahman y Hamada (2019) afirmaron que los hospitales producen grandes cantidades de información de manera diaria lo que genera un gran reto en su almacenamiento y su transmisión a través de su limitado ancho de banda por el internet. Debido a esto es que hay un incremento en la demanda para la investigación de compresión de datos y en teoría de la comunicación para solventar dichos retos.

Además, Alshorman (2020) añadió que las imágenes médicas han ganado un rol importante en los sistemas de hospitales y que guardar y enviar imágenes médicas de gran tamaño es un reto para los sistemas de administración de los hospitales. Las imágenes médicas de gran resolución obtenidas de distintas fuentes (Rayos X, Mamografías, etc.) requieren de una mayor capacidad de almacenamiento y ancho de banda.

Debido a la realidad problemática expuesta esta investigación se justificó de manera teórica, tecnológica y económicamente. Por el lado de la justificación teórica Palanisamy et al. (2018) propusieron un método para la compresión de imágenes usando los algoritmos LZW y Codificación aritmética. Alshorman (2020) usó el algoritmo RLE para la compresión de imágenes en aplicaciones de telemedicina.

Para la justificación tecnológica Palanisamy et al. (2018) mencionaron que la combinación de los algoritmos LZW y Codificación aritmética presento mejor calidad y compresión en la compresión de imágenes multiespectrales satelitales. Poolakkachalil Thafseela Koya y Chandran Saravanan (2018) indicaron que entre la compresión estereoscópica usando Codificación aritmética y la compresión estereoscópica usando Codificación Huffman la que dio mejor resultado respecto al tiempo de ejecución fue la codificación aritmética y se puede usar en áreas donde la calidad sea un factor importante.

Por otro lado, en la justificación económica Rahman y Hamada (2019) indicaron que es un gran reto transmitir imágenes grandes a través del internet con un ancho de banda reducido. Mbewe Phyela y D. Asare Sampson (2017) mencionaron que es un problema transmitir grandes imágenes por la red debido a los grandes tiempos de carga y eso consume ancho de banda.

En base a la problemática presentada, se plantearon el problema general y los problemas específicos de esta investigación. Siendo el problema general ¿Qué algoritmo presenta la mejor tasa de compresión, tiempo de compresión y descompresión en imágenes entre los algoritmos IBK, LZW y RLE? Siendo los problemas específicos:

- PE1: ¿Qué algoritmo presenta la mejor tasa de compresión de imágenes entre los algoritmos IBK, LZW y RLE?
- PE2: ¿Qué algoritmo presenta el mejor tiempo de compresión de imágenes entre los algoritmos IBK, LZW y RLE?
- PE3: ¿Qué algoritmo presenta el mejor tiempo de descompresión de imágenes entre los algoritmos IBK, LZW y RLE?

El objetivo general será determinar qué algoritmo presenta la mejor tasa de compresión, tiempo de compresión y descompresión en imágenes entre los algoritmos IBK, LZW y RLE. Siendo los objetivos específicos los siguientes

- OE1: Determinar qué algoritmo presenta la mejor tasa de compresión de imágenes entre los algoritmos IBK, LZW y RLE.
- OE2: Determinar qué algoritmo presenta el mejor tiempo de compresión de imágenes entre los algoritmos IBK, LZW y RLE.
- OE3: Determinar qué algoritmo presenta el mejor tiempo de descompresión de imágenes entre los algoritmos IBK, LZW y RLE.

Siendo la hipótesis general: “El algoritmo IBK tendrá mejor tasa de compresión, tiempo de compresión y descompresión en imágenes comparado con LZW y RLE.”

- HE1: El algoritmo IBK tendrá mejor tasa de compresión, en imágenes comparado con LZW y RLE.

  Palanisamy et al. (2018) mencionaron que el radio de compresión está definido como el número de bits de la representación del tamaño de la imagen original entre el número de bits a representar de la imagen comprimida. Asimismo, Rahman y Hamada (2019) indicaron que un algoritmo de compresión de imágenes es mejor cuando tiene más radio de compresión.

- HE2: El algoritmo IBK tendrá menor tiempo de compresión en imágenes comparado con LZW y RLE.

  Rahman, Hamada y Shin (2021) indicaron que el tiempo de compresión es una de las principales maneras para medir un algoritmo de compresión de imágenes. Gupta, Bansal and Khanduja (2017) evaluaron su algoritmo midiendo la velocidad de compresión.

- HE3: El algoritmo IBK tendrá menor tiempo de descompresión en imágenes comparado con LZW y RLE.

  Rahman, Hamada y Shin (2021) indicaron que el tiempo de descompresión se usó para medir el algoritmo de compresión de imágenes. Gupta, Bansal y Khanduja (2017) mencionaron que evaluaron su algoritmo midiendo la velocidad de descompresión.

II. MARCO TEÓRICO

En este capítulo se muestran investigaciones previas o antecedentes internacionales y posteriormente también se detallan las teorías relacionadas similares a esta investigación.

Liu et al. (2022) propusieron un algoritmo para compresión de imágenes combinando la predicción linear, el integer wavelet transform y Codificación Huffman. Liu et al. (2022) aplicaron el algoritmo creado a tres diferentes sets de imágenes, usando los Bits per-pixel como medición absoluta para la ratio de compresión. El algoritmo logró reducir entre un 6.22% y un 72.36% el tamaño de las imágenes indicando que su algoritmo es más apropiado para imágenes con una textura compleja y gran resolución. Liu et al. (2022) también indicaron que como trabajo a futuro aumentarán el espectro de imágenes enfocado a una menor escala de colores ya que el algoritmo se centra solo en imágenes naturales de 256 colores. Liu et al. (2022) señalan que para un futuro trabajo para mejorar su algoritmo se buscaría implementarlo en un lenguaje de programación como Python siendo este el lenguaje utilizado en este estudio.

Yousif y Salman (2021) aplicaron la codificación aritmética con la transformada de coseno discreta con el modelo de color YCbCr a imágenes a color. Yousif y Salman (2021) aplicaron su algoritmo a un set de imágenes en formato BMP de 512x512px implementándolo en MATLAB. Como resultados concluyeron que aplicando la codificación aritmética solo obtuvo un 1.14% obteniendo una mejor compresión aplicando su algoritmo en 5.5%. Además, como trabajo a futuro propone agregar otras transformaciones como transformación de onda, etc. para obtener un mejor radio de compresión. Yousif y Salman (2021) definen los parámetros para medir los resultados obtenidos en las pruebas, entre ellos el radio de compresión o tasa de compresión que fue utilizado en este estudio.

Alshorman (2020) propuso una técnica utilizando una versión modificada del algoritmo RLE añadiendo la técnica del escaneo Bloque-Bloque aplicado a imágenes médicas en blanco y negro. Alshorman (2020) implementó su algoritmo en MATLAB con un set de imágenes médicas obtenidas de muestra y comparándolo con otros algoritmos para la compresión de imágenes médicas. Alshorman (2020) obtuvo un radio de compresión de 2.8 comparado a los otros algoritmos que hicieron menos. Como trabajo futuro Alshorman (2020) recomendó que se realice una comparación de su algoritmo a imágenes médicas a color RGB.

Alshorman (2020) mencionó que el algoritmo RLE es considerado un algoritmo de rápida compresión comparado con otros algoritmos más complejos. Es por ese motivo que se decidió utilizar el algoritmo RLE para este proyecto.

Surya Teja Kodukulla (2020) llevó a cabo un estudio comparativo entre los algoritmos LZW, RLE, Huffman, DCT en modo sin perdidas y DWT. Surya Teja Kodukulla (2020) usó MATLAB para la implementación de los algoritmos con un set de imágenes por defecto. El resultado obtenido es que el algoritmo LZW produjo imágenes en binario siendo incapaz de generar una imagen sin perdida, mientras que los algoritmos de Huffman y RLE obtuvieron resultados similares en la compresión de imágenes en un rango de 2.5 a 3.7, por otro lado, los algoritmos DCT y DWT obtuvieron un rango de entre 2 a 3.5. Surya Teja Kodukulla (2020) concluyó que el algoritmo DWT es el más eficiente debido al método que usa el algoritmo para comprimir imágenes. Como trabajo futuro se menciona que se podría mejorar el algoritmo DWT refinando ciertos parámetros del algoritmo. Surya Teja Kodukulla (2020) mencionó que el algoritmo RLE y el algoritmo Huffman no son estables para la compresión de imágenes grandes ya que por lo general el algoritmo falla en estos casos. Es por esto por lo que para este estudio se decidió utilizar imágenes de resolución baja.

Salman y Hassan (2019) crearon un algoritmo combinando el algoritmo RLE con el algoritmo LZW y VLC para comprimir imágenes de resonancia magnética. Salman y Hassan (2019) aplicaron su algoritmo con un set de imágenes obtenidas del scanner Siemens MAGNETOM utilizando como lenguaje de programación a C#. Salman y Hassan (2019) concluyeron que la mejor opción para este tipo de imágenes es el algoritmo RLE y que la aplicación de LZW no tuvo mucho impacto en la compresión debido en la parte del proceso en que fue usado, pero de todas formas se obtuvo una mejor compresión en un 25%. Es asi que para este estudio también se decidió usar un lenguaje de programación de escritorio y usar los algoritmos RLE y LZW para demostrar su eficacia.

Yuan y Hu (2019) realizaron un análisis del estándar JPEG2000 que se basa en la codificación de Huffman y la transformación en ondícula para mostrar las diferentes distorsiones que se consiguen al aplicar distintos radios de compresión. Yuan y Hu (2019) realizaron la experimentación comparando los radios de una misma imagen en formato JPEG y JPEG2000. Yuan y Hu (2019) concluyeron que

el formato JPEG2000 mejora al formato JPEG tradicional ya que no muestra el efecto “cuadrado” que presenta el formato JPEG al ser sometido a una mayor compresión.

Saidani, Xiang y Mansouri (2019) crearon un nuevo algoritmo utilizando al algoritmo de compresión de imágenes RLE y el algoritmo BZip2 para Redes de sensores inalámbricos (WSN). Saidani, Xiang y Mansouri (2019) utilizaron múltiples imágenes obtenidas de los sensores descargados de distintos sitios que proveen estas imágenes y compararon su algoritmo con los algoritmos GZIP, BZIP2, RAR, LZMA, LZ77 y LZW. Saidani, Xiang y Mansouri (2019) obtuvieron buenos resultados en materia de compresión de datos y obtuvieron un rendimiento estable para con su algoritmo mientras que los algoritmos comparados no presentaron un rendimiento uniforme. Como trabajo a futuro propusieron resaltar mejor los datos en bruto obtenidos del sensor para su posterior implementación dentro del mismo. Gracias a esto Saidani, Xiang y Mansouri (2019) es que en nuestro estudio también se utilizó el algoritmo RLE combinándolo con el algoritmo LZW.

Birajdar Amit et al. (2019a) realizaron una mejora al algoritmo combinando varios métodos que mejoran al algoritmo RLE para eliminar el principal problema de este que es devolver datos mayores en peso que el ingresado en ciertos tipos de imágenes, en este caso imágenes en vertical y horizontal en blanco y negro o a color. Birajdar Amit et al. (2019a) concluyeron que después de aplicar los distintos métodos que buscan mejorar el algoritmo RLE se logró obtener un algoritmo mejorado que es más inteligente y robusto al momento de procesar los datos. Es asi que debido al uso del algoritmo RLE es que se también se utiliza en este estudio imágenes en blanco y negro o en escala de grises.

Fathahillah y Zain (2019) crearon un algoritmo para comprimir imágenes homogéneas utilizando el algoritmo de compresión de imágenes Shannon-Fano. Para esto, Fathahillah y Zain (2019) utilizaron MATLAB para la aplicación de su algoritmo y utilizaron imágenes obtenidas de una cámara web teniendo como resultado una compresión de alrededor del 52% pero aclararon que la aplicación de este algoritmo es óptima con imágenes de un tamaño especifico que en este estudio fueron imágenes de 160x120. Es debido a esto que en nuestro estudio también se utilizan imágenes con un tamaño fijo.

Birajdar Amit et al. (2019b) hicieron una comparación de los algoritmos LZW y RLE para comprimir imágenes. Birajdar Amit et al. (2019b) utilizaron un set de imágenes especificas con ciertos patrones para medir el rendimiento de estos algoritmos. Birajdar Amit et al. (2019b) concluyeron que el algoritmo RLE por sí solo no fue tan eficiente con el set de imágenes y que hicieron la prueba también con un algoritmo RLE mejorado, siendo este el que obtuvo mejores resultados, pero que a pesar de esto el algoritmo LZW tuvo mejor radio de compresión que ambos, pero la compresión tomaba más tiempo. Debido a la mejor tasa de compresión obtenida por el algoritmo LZW es que se decidió usarlo en nuestro estudio.

Akinyemi Omololu y Modinat Abolore (2019) propusieron un método para la compresión de imágenes que combinaba Transformación de Longitud de onda discreta y el algoritmo de compresión LZW. Akinyemi Omololu y Modinat Abolore (2019) implementaron el algoritmo en MATLAB 2016A usando tipo de filtro 1.3 y 1.5 bior y para cada imagen el rendimiento fue medido usando la función peak signal noise ratio, también aplico a los Bit per-pixel como medida absoluta para representar el numero en promedio de bits necesitados para codificar cada imagen y el radio de compresión. Como resultado obtuvo una mejora en la compresión de imágenes de 1.58% utilizando el filtro bior 1.3. Debido a que aquí nuevamente se obtiene un resultado favorable con el algoritmo LZW es que se le consideró para su uso en nuestro estudio.

Palanisamy et al. (2018) propusieron un método hibrido usando LZW y Codificación aritmética para la compresión de imágenes multiespectrales. Palanisamy et al. (2018) compararon su algoritmo con métodos de compresión como la codificación Huffman, RLE, LZW y Codificación aritmética utilizando imágenes de la web del USGS (United States Geological Survey) concluyendo que su algoritmo hibrido dio mejores resultados que los algoritmos antes mencionados de manera individual, tanto como en calidad de imagen y factores de compresión. Gracias a los buenos resultados obtenidos por la combinación de los algoritmos RLE y LZW en imágenes satelitales es que se considera nuevamente su uso para nuestro estudio.

Alam et al. (2018) combinaron el algoritmo LZW con computación paralela y mencionan que los tiempos de compresión se redujeron, pero a escala de milisegundos manteniendo la naturaleza sin pérdida de las imágenes. Alam et al.

(2018) utilizaron la computación paralela ofrecida por la arquitectura CUDA de las tarjetas de video de NVidia para comprimir las imágenes usando los recursos tanto como del CPU como del GPU. Alam et al. (2018) obtuvieron una reducción de hasta el 25% en el tiempo de compresión de imágenes dependiendo del tipo de imagen, estando su algoritmo desarrollado con Visual Studio 2010, con CUDA 5.5 y lenguaje C. Alam et al. (2018) mencionó que a futuro considerarían usar más hilos para el proceso de computación en la aplicación del algoritmo para asi reducir más el tiempo de compresión. Debido a que Alam et al. (2018) decide usar un lenguaje de programación de escritorio para la implementación de su algoritmo es que se decide también usar un lenguaje de escritorio para nuestro estudio.

Mbewe Phyela y D. Asare Sampson (2017) llevaron a cabo un análisis comparativo entre los algoritmos de Huffman y Codificación aritmética. Mbewe Phyela y D. Asare Sampson (2017) utilizaron la aplicación CMedia Compressor para aplicar los algoritmos tanto de Huffman como el de Codificación aritmética a imágenes recolectadas por ellos mismos de distintas fuentes y que tengan cierto acercamiento a imágenes que utilizaría el usuario común. Mbewe Phyela y D. Asare Sampson (2017) concluyeron que el algoritmo de codificación aritmética tuvo mejores resultados para todos los distintos tipos de imágenes de muestra en términos de ahorro de espacio. Al igual que Mbewe Phyela y D. Asare Sampson (2017) utilizaron un software para poder probar los algoritmos nuestro estudio también hace uso de un software para probar los algoritmos.

Sangeetha, Betty y Nanda (2017) utilizaron los algoritmos LZW y un híbrido de LZW para la compresión de imágenes biométricas de lectura de iris. Sangeetha, Betty y Nanda (2017) para llevar a cabo la implementación de estos algoritmos usaron MATLAB y los aplicaron de forma secuencial en la compresión de imágenes para reducir la ocurrencia de ruido, comparando los resultados con la Codificación Huffman y el algoritmo LZW. Sangeetha, Betty y Nanda (2017) concluyeron que su algoritmo presentó mejor rendimiento comparado con Huffman y LZW. Sangeetha, Betty y Nanda (2017) como futuro trabajo indicaron que es posible expandir su algoritmo para el procesamiento de imágenes de otros dispositivos biométricos como huellas digitales e imágenes de la palma de la mano completa. Debido a los buenos resultados obtenidos por el algoritmo LZW en este estudio es que se consideró al algoritmo LZW en nuestro estudio.

Sangeetha y Betty (2017) llevaron a cabo un análisis comparativo entre algoritmos de compresión de imágenes como RLE, LZW, Codificación Huffman, Codificación Delta, etc. En imágenes biométricas. Sangeetha y Betty (2017) concluyen que el algoritmo LZW obtiene mejor resultados en rendimiento para comprimir imágenes biométricas. Como futuro trabajo, Sangeetha y Betty (2017) propusieron expandir este análisis a imágenes de altas dimensiones. Debido a este análisis es que se decide utilizar el algoritmo LZW en nuestro estudio

Pero para poder tener un mejor entendimiento de estos trabajos previos, es necesario conocer algunos conceptos que son fundamentales para la compresión de datos y específicamente para la compresión de imágenes.

De manera que, Senthilkumaran et al. (2017) definieron que la compresión es una forma de reducir el tamaño de los archivos. Unos ejemplos de extensiones de archivos que comprimen archivos son: .sit, .tar, .zip.

Mientras que, Alshorman (2020) mencionó que la descompresión reconstruye el archivo original desde un estado de compresión.

Por consiguiente, Sarathe et al. (2017) definieron a la compresión de datos como el proceso de convertir un flujo de datos de entrada en otro flujo de datos pero que tenga menor tamaño. Senthilkumaran et al. (2017) también lo definen como un método que reduce el tamaño de los archivos.

En base a lo anterior Mahdi y Hassan (2017) mencionaron que la compresión de datos se clasifica en dos tipos: sin pérdida y con pérdida.

Mahdi y Hassan (2017) precisaron que la compresión sin pérdidas es un método por el cual tiene el potencial para reestablecer de manera precisa el flujo de datos a partir de la data comprimida. Fitriya, Purboyo y Prasasti (2017) definieron a la compresión sin pérdidas es el proceso de convertir la data original con data comprimida que es más concisa sin reducir la pérdida de información. Para finalizar, Sharma (2018) indicó que en la compresión sin pérdida la imagen reconstruida, después de haberla comprimido es numéricamente idéntica a la imagen original.

Por otra parte, Senthilkumaran et al. (2017) mencionaron que para la compresión de imágenes con pérdida habrá posibilidad de pérdida de información, pero debería estar bajo el límite de tolerancia. Sarathe et al. (2017) añadieron que en la compresión con pérdida reduce el tamaño de los archivos al eliminar algunos datos que no son necesarios y que no serán reconocidos por el ser humano

después de la compresión. Khandwani y Ajmire (2018) precisaron que la compresión con pérdida es frecuentemente usada para comprimir archivos multimedia. Palanisamy et al. (2018) define la compresión con perdida a que en el proceso de reconstrucción de la imagen hay cierta perdida de datos.

Definiendo al algoritmo de compresión RLE (Run Length Encoding), Al-Jawaherry y Hamid (2021) mencionaron que es un método de compresión sin perdidas que se basa en la ocurrencia de datos en vez de datos estadísticos usando un dúo de longitud – valor para reemplazar los datos en donde el valor es el dato en sí y longitud el número de repeticiones. Khandwani y Ajmire (2018) describe al algoritmo RLE como uno de los más simples de los métodos de compresión, siendo el principio de este algoritmo en explotar los valores repetidos de un archivo. Esta repetición de la cadena de caracteres es llamada un recorrido y el algoritmo cuenta la cantidad de repeticiones de cada símbolo y los usa para representar el recorrido,

Gopinath y Ravisankar (2020) definieron al algoritmo LZW (Lempel-Ziv-Welch) como un algoritmo de compresión sin perdidas basado en diccionarios en donde busca patrones recurrentes en los datos para reclamar espacio. Arora y Kumar (2018) menciona que el algoritmo durante la compresión forma una tabla con la codificación de los datos y también con la decodificación de estos. Adicionalmente, Gopinath y Ravisankar (2020) menciona que el algoritmo LZW realiza la compresión reemplazando las cadenas de un carácter con un código, luego el algoritmo empieza a leer la data agrupando cada símbolo en cadenas para al final convertir estas cadenas en códigos. Este método usa una tabla de 4096 códigos de los cuales del 0 al 256 son entradas fijas, conforme la compresión avanza el algoritmo agrega a esta tabla los grupos de caracteres que no encuentre entre las 256 iniciales.

Palanisamy et al. (2018) define a la tasa de compresión como el número de bits que representan el tamaño original de la imagen entre el número de bits de la representación de la imagen comprimida. Y se obtiene a través de la formula:

$$CR = T1 / T2$$

T1 = Numero de bits de la imagen original

T2 = Numero de bits de la imagen comprimida

Rahman, Hamada y Shin (2021) mencionan que el tiempo de compresión y el tiempo de descompresión son los tiempos requeridos para codificar y decodificar una imagen.

Lozano et al. (2018) definen como a Scrum como uno de los marcos de trabajo ágil más reciente con la finalidad de producir software en corto tiempo para reducir costos de producción sin sacrificar las demandas de los clientes. Mas sobre la metodología Scrum ver Anexo 5.

III. METODOLOGÍA

Para este capítulo se detallarán aspectos de la investigación como el tipo y diseño de la investigación, también se especifican los criterios para la selección de la población además de los aspectos éticos.

3.1 Tipo y diseño de investigación

Para este proyecto de investigación el tipo de investigación es aplicada. Teodoro Nicomedes (2018) indicó que este tipo de investigación se enfoca en resolver problemas que puedan presentar procesos o servicios en la actividad humana.
Se escogió este tipo de investigación porque al hacer realizar un algoritmo de compresión de imágenes se busca dar solución a un problema que se tiene al momento de almacenar y transmitir imágenes.

Siendo esta investigación de enfoque cuantitativo ya que nos permite obtener datos numéricos relacionadas a ciertas variables. Damián Cabezas Mejía y Andrade Naranjo Johana Torres Santamaría (2018) indicaron que el este método recolecta datos numéricos para probar una hipótesis e identificar patrones de comportamiento y avalar teorías.

El diseño de esta investigación fue preexperimental, porque se cuenta con un grupo existente aplicándole un tratamiento experimental y se miden los efectos causados. Damián Cabezas Mejía y Andrade Naranjo Johana Torres Santamaría (2018) mencionaron que estas investigaciones no tienen algún tipo de control y que solo se analiza una sola variable.

3.2 Variables y operacionalización

Para este proyecto de investigación la variable de estudio fue el efecto del algoritmo de compresión basado en LZW y RLE. En el Anexo 1 se adjunta la matriz de operacionalización. A continuación, se detalla lo siguiente:

a. Definición Conceptual: La compresión de imágenes son un conjunto de técnicas que son aplicadas a las imágenes para después guardarlas de manera efectiva. Qasim, Din y Alyousuf (2020)
b. Definición Operacional: Técnicas de compresión de imágenes son mejores cuando contienen menos código en promedio, tiempo de compresión y

descompresión y da un mejor radio de compresión. Rahman y Hamada (2019)

c. Dimensiones:
    - Compresión de imágenes. Boopathiraja, Kalavathi y Dhanalakshmi (2019)
    - Descompresión de imágenes. Rahman, Hamada y Shin (2021)
d. Indicadores:
    - Tasa de compresión de imágenes. Boopathiraja, Kalavathi y Dhanalakshmi (2019)
    - Tiempo de compresión. Rahman, Hamada y Shin (2021)
    - Tiempo de descompresión. Gupta, Bansal y Khanduja (2017)

3.3 Población, muestra y muestreo

A continuación, se detalla los conceptos asociados a población, muestra, muestreo y unidad de análisis:

- **Población:**

Hernández Sampieri, Fernández Collado y Baptista Lucio (2014) señalan que la población es un conjunto de casos que cuentan con ciertas especificaciones.

Por lo tanto, la población para esta investigación serán 320 archivos de imágenes de dispositivos biométricos. Siendo los siguientes los criterios de inclusión y de exclusión:

- Criterios de inclusión: Archivos de imágenes de dispositivos biométricos.
- Criterios de exclusión: Otros tipos de archivos que no sean imágenes y que ya hayan sido procesados o comprimidos.

- **Muestra:**

Hernández Sampieri, Fernández Collado y Baptista Lucio (2014) indica que la muestra es un subgrupo de la población del cual se extraerán datos. Siendo que la muestra de tipo no probabilística para esta investigación será de 175 imágenes.

- **Muestreo**

El muestreo para esta investigación será de tipo probabilístico. Cruz del Castillo, Olivares Orozco y González García (2014) menciona que el muestreo probabilístico

se da cuando todos los integrantes de la población tienen la misma probabilidad para ser seleccionados.

3.4 Técnicas e instrumentos de recolección de datos

Para este proyecto de investigación la técnica de recolección de datos fue la de la observación. Hernández Sampieri, Fernández Collado y Baptista Lucio (2014) mencionó que la observación como recolección de datos consiste en el registro sistemático, válido y confiable de comportamientos en un conjunto de categorías. Para ello se contó con un instrumento de recolección de datos en formato Excel en el cual estuvieron los indicadores descritos para esta investigación con la respectiva validez de contenido. Hernández Sampieri, Fernández Collado y Baptista Lucio (2014) indicó que un instrumento ideal es el cual registra los datos observables que representan las variables a investigar.

3.5 Procedimientos

En este proyecto de investigación se utilizó un método de análisis de datos en los cuales se escogieron 175 archivos del conjunto de datos de iris de la Multimedia University (MMU2), estos archivos son de uso libre. Para la manipulación de los datos se creó un nuevo algoritmo hibrido que combina los algoritmos RLE y LZW y se comparó con los algoritmos LZW y RLE por separado. Para la medición de la tasa de compresión, tiempo de compresión y tiempo de descompresión se realizó un aplicativo en el lenguaje de programación Python que se encargará del procesamiento de las imágenes. Para demostrar que el algoritmo cumplió con su objetivo, se siguieron los siguientes pasos:

- Identificar la falta de un algoritmo que comprima de imágenes biométricas sin pérdida.
- Seleccionar una metodología adecuada para el desarrollo del software.
- Obtener la implementación en el lenguaje seleccionado de los algoritmos que se usaron como base
- Desarrollar prototipos de la interfaz del software.
- Implementar los algoritmos dentro del software.
- Implementar en código los aspectos a medir de los algoritmos
- Realizar pruebas para medir el nivel de eficacia de los algoritmos
- Realizar el instrumento para llevar un registro de los resultados obtenidos.

- Se registraron los resultados del procesamiento de imágenes para medir el tiempo de compresión, el tiempo de descompresión y radio de compresión.

3.6 Método de análisis de datos

Para este proyecto de investigación se efectuará un análisis estadístico descriptivo. Luego se usará IBM SPSS para el procesamiento de los datos obtenidos. Para las pruebas de normalidad se usó la prueba de Kolmogórov-Smirnov ya que en esta investigación el tamaño de muestra será de 150 imágenes biométricas.

- **Kolmogórov-Smirnov**

Cruz del Castillo, Olivares Orozco y González García (2014) indicaron que esta prueba busca conocer un grado de acuerdo con la distribución de un conjunto de valores muestreados y una distribución teórica especifica.

Por otro lado, en las pruebas de promedios se utilizará como prueba no paramétrica a la prueba de Wilcoxon.

- **Prueba de Wilcoxon**

Cruz del Castillo, Olivares Orozco y González García (2014) mencionaron que esta prueba calcula la diferencia entre dos variables y clasifica sus diferencias como positivas, negativas o empatadas. Siendo el caso que tengan una distribución similar el número de diferencias positivas y negativas no difieren de forma significativa.

3.7 Aspectos éticos

Para la realización de esta investigación se tuvieron en cuenta ciertos principios éticos. Tales como el respeto a la autoría de las fuentes de información al citar cada referencia con el estilo ISO-690. También esta investigación cumplió con el código de ética de la universidad César Vallejo asi como también del código de ética del Colegio de Ingenieros del Perú con sus árticos 37, 42 y 44.

IV. RESULTADOS

En este capítulo se ahondan en los resultados obtenidos dentro de la investigación de acuerdo con los indicadores de tasa de compresión, tiempo de compresión y tiempo de descompresión.

4.1. Datos Descriptivos

A continuación, se detallan los estadísticos descriptivos además de las pruebas realizadas en los indicadores.

4.1.1. Tasa de compresión

El resultado de los estadísticos descriptivos para la tasa de compresión se muestra en la tabla 1.

Estadísticos descriptivos				
		tasa_compresion_rle	tasa_compresion_lzw	tasa_compresion_ibk
N	Válido	175	175	175
	Perdidos	0	0	0
Media		0,252660861722600	0,323007087013374	0,513771753474031
Mediana		0,252676758965650	0,284781118767835	0,447198332795210
Desv. estándar		0,000189148917254	0,081937117696029	0,137461509044145
Rango		0,0009223768271758	0,3963650588798794	0,6863449711259444
Mínimo		0,2522784474764870	0,2522987788789444	0,3983996652894723
Máximo		0,2532008243036628	0,6486638377588237	1,0847446364154167

Tabla 1: Estadísticos descriptivos – Tasa de compresión

En los estadísticos descriptivos para la tasa de compresión se muestran los datos distribuidos de los distintos algoritmos comparados. En este caso, el algoritmo IBK obtuvo un promedio (media) mayor, seguido del algoritmo LZW, siendo el algoritmo RLE el de menor promedio.

Prueba de normalidad

Para la prueba de normalidad de la tasa de compresión se utilizó la prueba de Kolmogórov-Smirnov ya que la muestra es de 175 archivos de imágenes. Los resultados se pueden observar en la tabla 2.

Pruebas de normalidad						
	Kolmogórov-Smirnov			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
tasa_compresion_rle	0,056	175	0,200*	0,981	175	0,019
tasa_compresion_lzw	0,208	175	0,000	0,777	175	0,000
tasa_compresion_ibk	0,213	175	0,000	0,763	175	0,000
*. Esto es un límite inferior de la significación verdadera.						

a. Corrección de significación de Lilliefors

Tabla 2: Prueba de normalidad – Tasa de compresión

En este caso algunos valores de significancia son menores de 0.05, por lo tanto, los indicadores no se ajustan a una distribución normal.

4.1.2. Tiempo de compresión

El resultado de los estadísticos descriptivos para el tiempo de compresión se muestra en la tabla 3.

Estadísticos descriptivos				
		tiempo_compresion_rle	tiempo_compresion_lzw	tiempo_compresion_ibk
N	Válido	175	175	175
	Perdidos	0	0	0
Media		0,194884372000275	0,090071055428229	0,092324366856982
Mediana		0,191409900002327	0,091627400001016	0,090837799994915
Desv. estándar		0,015818275514102	0,009924258173283	0,013240611218551
Rango		0,1087532000019564	0,0523436000003130	0,0695288999995682
Mínimo		0,1813728999986779	0,0677647999982582	0,0663308000002871
Máximo		0,2901261000006343	0,1201083999985713	0,1358596999998554

Tabla 3: Estadísticos descriptivos – Tiempo de compresión

En los estadísticos descriptivos para la tasa de compresión se muestran los datos distribuidos de los distintos algoritmos comparados. En este caso, el algoritmo RLE obtuvo un promedio (media) mayor, seguido del algoritmo IBK, siendo el algoritmo LZW el de menor promedio.

Prueba de normalidad

Para la prueba de normalidad de la tasa de compresión se utilizó la prueba de Kolmogórov-Smirnov ya que la muestra es de 175 archivos de imágenes. Los resultados se pueden observar en la tabla 4.

Pruebas de normalidad						
	Kolmogórov-Smirnov			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
tiempo_compresion_rle	0,225	175	0,000	0,578	175	0,000
tiempo_compresion_lzw	0,074	175	0,022	0,983	175	0,033
tiempo_compresion_ibk	0,051	175	0,200*	0,984	175	0,044
*. Esto es un límite inferior de la significación verdadera.						
a. Corrección de significación de Lilliefors						

Tabla 4: Prueba de normalidad – Tiempo de compresión

En este caso algunos valores de significancia son menores de 0.05, por lo tanto, los indicadores no se ajustan a una distribución normal.

4.1.3. Tiempo de descompresión

El resultado de los estadísticos descriptivos para el tiempo de descompresión se muestra en la tabla 5.

Estadísticos descriptivos				
		tiempo_descompresion_rle	tiempo_descompresion_lzw	tiempo_descompresion_ibk
N	Válido	175	175	175
	Perdidos	0	0	0
Media		0,110920113142804	0,026731789142941	0,067669688571394
Mediana		0,110934900003485	0,028485700000601	0,072678600001382
Desv. estándar		0,002907289991950	0,006388756428589	0,014101596223755
Rango		0,0171469999986584	0,0298205999970378	0,0667048000032082
Mínimo		0,1052052000013646	0,0114420000027167	0,0288708000007318
Máximo		0,1223522000000230	0,0412625999997545	0,0955756000039401

Tabla 5: Estadísticos descriptivos – Tiempo de descompresión

En los estadísticos descriptivos para el tiempo de descompresión se muestran los datos distribuidos de los distintos algoritmos comparados. En este caso, el algoritmo RLE obtuvo un promedio (media) mayor, seguido del algoritmo IBK, siendo el algoritmo LZW el de menor promedio.

Prueba de normalidad

Para la prueba de normalidad de la tasa de compresión se utilizó la prueba de Kolmogorov-Smirnov ya que la muestra es de 175 archivos de imágenes. Los resultados se pueden observar en la tabla 6.

Pruebas de normalidad						
	Kolmogórov-Smirnov			Shapiro-Wilk		
	Estadís-tico	gl	Sig.	Estadís-tico	gl	Sig.
tiempo_descompresion_rle	0,048	175	0,200*	0,963	175	0,000
tiempo_descompresion_lzw	0,127	175	0,000	0,951	175	0,000
tiempo_descompresion_ibk	0,159	175	0,000	0,919	175	0,000
*. Esto es un límite inferior de la significación verdadera.						
a. Corrección de significación de Lilliefors						

Tabla 6: Pruebas de normalidad – Tiempo de descompresión

En este caso algunos valores de significancia son menores de 0.05, por lo tanto, los indicadores no se ajustan a una distribución normal.

4.2. Prueba de Hipótesis

En este apartado se darán detalles sobre las pruebas de hipótesis realizadas a cada indicador.

4.2.1. Prueba de Hipótesis 1

HE1$_0$: El algoritmo IBK no tuvo mejor tasa de compresión que los algoritmos LZW y RLE.

HE1$_1$: El algoritmo IBK tuvo mejor tasa de compresión que los algoritmos LZW y RLE.

Prueba de Wilcoxon

A continuación, en la tabla 7, se muestran los resultados de la prueba de Wilcoxon sobre la tasa de compresión entre el algoritmo IBK y RLE.

Rangos				
		N	Rango promedio	Suma de rangos
tasa_compresion_ibk - tasa_compresion_rle	Rangos negativos	0[a]	0,00	0,00
	Rangos positivos	175[b]	88,00	15400,00
	Empates	0[c]		
	Total	175		
a. tasa_compresion_ibk < tasa_compresion_rle				
b. tasa_compresion_ibk > tasa_compresion_rle				
c. tasa_compresion_ibk = tasa_compresion_rle				

Tabla 7: Prueba de Wilcoxon – Tasa de compresión algoritmo IBK y algoritmo RLE

En la tabla 8 se muestran los resultados de los estadísticos de la prueba de Wilcoxon sobre la tasa de compresión entre el algoritmo IBK y RLE.

Estadísticos de prueba[a]	
	tasa_compresion_ibk - tasa_compresion_rle
Z	-11,473[b]
Sig. asin. (bilateral)	0,000
a. Prueba de rangos con signo de Wilcoxon	
b. Se basa en rangos negativos.	

Tabla 8: Prueba de hipótesis – Tasa de compresión algoritmo IBK y algoritmo RLE

Prueba de Wilcoxon

A continuación, en la tabla 9, se muestran los resultados de la prueba de Wilcoxon sobre la tasa de compresión entre el algoritmo IBK y LZW.

Rangos				
		N	Rango promedio	Suma de rangos
tasa_compresion_ibk - tasa_compresion_lzw	Rangos negativos	0[a]	0,00	0,00
	Rangos positivos	175[b]	88,00	15400,00
	Empates	0[c]		
	Total	175		
a. tasa_compresion_ibk < tasa_compresion_lzw				
b. tasa_compresion_ibk > tasa_compresion_lzw				
c. tasa_compresion_ibk = tasa_compresion_lzw				

Tabla 9: Prueba de Wilcoxon – Tasa de compresión algoritmo IBK y algoritmo LZW

En la tabla 10 se muestran los resultados de los estadísticos de la prueba de Wilcoxon sobre la tasa de compresión entre el algoritmo IBK y LZW.

Estadísticos de prueba[a]	
	tasa_compresion_ibk - tasa_compresion_lzw
Z	-11,473[b]
Sig. asin. (bila-teral)	0,000
a. Prueba de rangos con signo de Wilcoxon	
b. Se basa en rangos negativos.	

Tabla 10: Prueba de hipótesis – Tasa de compresión algoritmo IBK y algoritmo LZW

Con los resultados obtenidos de la prueba de Wilcoxon, mostrados en las tablas 8 y 10, con significancia de 0.000 en ambos casos y estos al ser menores al 0.05 con 95% de confianza se acepta la hipótesis alternativa en donde el algoritmo IBK tuvo mejor tasa de compresión que los algoritmos RLE y LZW

4.2.2. Prueba de Hipótesis 2

HE1$_0$: El algoritmo IBK no tuvo mejor tiempo de compresión que los algoritmos LZW y RLE.

HE1$_1$: El algoritmo IBK tuvo mejor tiempo de compresión que los algoritmos LZW y RLE.

Prueba de Wilcoxon

A continuación, en la tabla 11, se muestran los resultados de la prueba de Wilcoxon sobre el tiempo de compresión entre el algoritmo IBK y RLE.

Rangos				
		N	Rango promedio	Suma de rangos
tiempo_compresion_rle - tiempo_compresion_ibk	Rangos negativos	0[a]	0,00	0,00
	Rangos positivos	175[b]	88,00	15400,00
	Empates	0[c]		
	Total	175		
a. tiempo_compresion_rle < tiempo_compresion_ibk				
b. tiempo_compresion_rle > tiempo_compresion_ibk				
c. tiempo_compresion_rle = tiempo_compresion_ibk				

Tabla 11: Prueba de Wilcoxon – Tiempo de compresión Algoritmo IBK y Algoritmo RLE

En la tabla 12 se muestran los resultados de los estadísticos de la prueba de Wilcoxon sobre la tasa de compresión entre el algoritmo IBK y RLE.

Estadísticos de prueba[a]	
	tiempo_compresion_rle - tiempo_compresion_ibk
Z	-11,473[b]
Sig. asin. (bila-teral)	0,000
a. Prueba de rangos con signo de Wilcoxon	
b. Se basa en rangos negativos.	

Tabla 12: Prueba de Hipótesis – Tiempo de compresión Algoritmo IBK y Algoritmo RLE

Prueba de Wilcoxon

A continuación, en la tabla 13, se muestran los resultados de la prueba de Wilcoxon sobre el tiempo de compresión entre el algoritmo IBK y LZW.

Rangos				
		N	Rango promedio	Suma de rangos
tiempo_compresion_lzw - tiempo_compresion_ibk	Rangos negativos	88[a]	101,34	8918,00
	Rangos positivos	87[b]	74,51	6482,00
	Empates	0[c]		
	Total	175		
a. tiempo_compresion_lzw < tiempo_compresion_ibk				
b. tiempo_compresion_lzw > tiempo_compresion_ibk				
c. tiempo_compresion_lzw = tiempo_compresion_ibk				

Tabla 13: Prueba de Wilcoxon – Tiempo de compresión Algoritmo IBK y Algoritmo LZW

En la tabla 14 se muestran los resultados de los estadísticos de la prueba de Wilcoxon sobre la tasa de compresión entre el algoritmo IBK y LZW.

Estadísticos de prueba[a]	
	tiempo_compresion_lzw - tiempo_compresion_ibk
Z	-1,815[b]
Sig. asin. (bila-teral)	0,070
a. Prueba de rangos con signo de Wilcoxon	
b. Se basa en rangos positivos.	

Tabla 14: Prueba de hipótesis – Tiempo de compresión Algoritmo IBK y Algoritmo LZW

Con los resultados obtenidos de la prueba de Wilcoxon, mostrados en las tablas 12 y 14, con significancia de 0.000 en el caso de la comparación entre el Algoritmo IBK y RLE y con significancia de 0.070 y estos al ser resultados disparejos y en el caso del primero ser menor que 0.05 con 95% de confianza y el segundo mayor, es de esta manera que se acepta la hipótesis nula en donde el algoritmo IBK no tuvo mejor tiempo de compresión que los algoritmos RLE y LZW.

4.2.3. Prueba de Hipótesis 3

HE1$_0$: El algoritmo IBK no tuvo mejor tiempo de descompresión que los algoritmos LZW y RLE.

HE1$_1$: El algoritmo IBK tuvo mejor tiempo de descompresión que los algoritmos LZW y RLE.

Prueba de Wilcoxon

A continuación, en la tabla 15, se muestran los resultados de la prueba de Wilcoxon sobre el tiempo de descompresión entre el algoritmo IBK y RLE.

Rangos				
		N	Rango promedio	Suma de rangos
tiempo_descompresion_rle - tiempo_descompresion_ibk	Rangos negativos	0[a]	0,00	0,00
	Rangos positivos	175[b]	88,00	15400,00
	Empates	0[c]		
	Total	175		
a. tiempo_descompresion_rle < tiempo_descompresion_ibk				
b. tiempo_descompresion_rle > tiempo_descompresion_ibk				
c. tiempo_descompresion_rle = tiempo_descompresion_ibk				

Tabla 15: Prueba de Wilcoxon – Tiempo de descompresión Algoritmo IBK y Algoritmo RLE

En la tabla 16 se muestran los resultados de los estadísticos de la prueba de Wilcoxon sobre la tasa de descompresión entre el algoritmo IBK y RLE.

Estadísticos de prueba[a]	
	tiempo_descompresion_rle - tiempo_descompresion_ibk
Z	-11,473[b]
Sig. asin. (bila-teral)	0,000
a. Prueba de rangos con signo de Wilcoxon	
b. Se basa en rangos negativos.	

Tabla 16: Prueba de hipótesis – Tiempo de descompresión Algoritmo IBK y Algoritmo RLE

Prueba de Wilcoxon

A continuación, en la tabla 17, se muestran los resultados de la prueba de Wilcoxon sobre el tiempo de descompresión entre el algoritmo IBK y LZW.

Rangos				
		N	Rango promedio	Suma de rangos
tiempo_descompresion_lzw - tiempo_descompresion_ibk	Rangos negativos	175[a]	88,00	15400,00
	Rangos positivos	0[b]	0,00	0,00
	Empates	0[c]		
	Total	175		
a. tiempo_descompresion_lzw < tiempo_descompresion_ibk				

b. tiempo_descompresion_lzw > tiempo_descompresion_ibk
c. tiempo_descompresion_lzw = tiempo_descompresion_ibk

Tabla 17: Prueba de Wilcoxon – Tiempo de descompresión Algoritmo IBK y Algoritmo LZW

En la tabla 18 se muestran los resultados de los estadísticos de la prueba de Wilcoxon sobre la tasa de descompresión entre el algoritmo IBK y RLE.

Estadísticos de prueba[a]	
	tiempo_descompresion_lzw - tiempo_descompresion_ibk
Z	-11,473[b]
Sig. asin. (bila-teral)	0,000
a. Prueba de rangos con signo de Wilcoxon	
b. Se basa en rangos positivos.	

Tabla 18: Prueba de hipótesis – Tasa de descompresión Algoritmo IBK y Algoritmo LZW

Con los resultados obtenidos de la prueba de Wilcoxon, mostrados en las tablas 16 y 18, con significancia de 0.000 en ambos casos y siendo menores que 0.05 con 95% de confianza, pero con un resultado diferente en la primera prueba es asi que se acepta la hipótesis nula en donde el algoritmo IBK no tuvo mejor tiempo de descompresión que los algoritmos RLE y LZW.

4.3. Resumen

A continuación, se muestran los resultados de aceptaciones o rechazos de las hipótesis de la investigación.

Cod.	Hipótesis	Condición
HE1	El algoritmo IBK tuvo mejor tasa de compresión que los algoritmos LZW y RLE	Aceptada
HE2	El algoritmo IBK tuvo menor tiempo de compresión que los algoritmos LZW y RLE	Rechazada
HE3	El algoritmo IBK tuvo menor tiempo de descompresión que los algoritmos LZW y RLE	Rechazada
HG	El algoritmo IBK tuvo mejor tasa de compresión, tiempo de compresión y tiempo de descompresión que los algoritmos LZW y RLE	Rechazada

Tabla 19: Tabla de resumen de hipótesis

V. DISCUSIÓN

En este capítulo se discuten los resultados obtenidos por esta investigación con los resultados obtenidos de los trabajos previos en lo que respecta a la tasa de compresión, tiempo de compresión y tiempo de descompresión. Además de realizarse un análisis más detallado de los resultados obtenidos en el capítulo anterior.

Para el apartado del tiempo de compresión de imágenes con los algoritmos IBK, LZW y RLE los tiempos que se obtuvieron en promedio fueron 0.0923, 0.0900 y 0.1948 en segundos respectivamente. Donde en base a estos resultados el algoritmo con menor tiempo de compresión de imágenes fue el algoritmo LZW, seguido por el algoritmo IBK, siendo el algoritmo RLE el algoritmo que obtuvo un tiempo de compresión mayor al resto.

En este sentido los resultados obtenidos en esta investigación difieren con los resultados obtenidos por Gopinath y Ravisankar (2020) en su estudio, en donde comparan los algoritmos de Delta encoding, RLE, Huffman y LZW Gopinath y Ravisankar (2020) destacaron que el algoritmo LZW obtiene un tiempo de compresión muy elevado (0.25s) comparado con los otros algoritmos especialmente con el algoritmo RLE (0.005s).

Adicionalmente, nuestro estudio difiere con el realizado por Birajdar Amit et al. (2019b) ya que, en este caso, se compararon los algoritmos RLE, LZW y una optimización del algoritmo RLE para comprimir imágenes y textos. Birajdar Amit et al. (2019b) obtuvieron unos resultados en los que el algoritmo RLE consiguió menos tiempo de compresión que el algoritmo LZW, la principal diferencia de nuestro estudio con éste es que la data ingresada para el procesado aparte de ser imágenes simples en blanco y negro, también fueron archivos de texto.

Asimismo, nuestro estudio también difiere del realizado por Chianphatthanakit, Boonsongsrikul y Suppharangsan (2018) ya que, en su estudio, proponen un algoritmo de compresión de imágenes y los comparan con los algoritmos LZW, RLE, GZIP y LEC, siendo en esta comparación en la cual el algoritmo RLE consiguió un tiempo menor que el algoritmo LZW. Una de las mayores diferencias de este estudio con el nuestro es que en este caso se procesan imágenes a colores de un data set con imágenes stock como lo son “city”, “mandrill”

y “Lenna” además de hacer una comparación con más algoritmos de compresión y utilizar MATLAB para la implementación de los algoritmos.

También Sampson D. Asare y Mbewe Phyela (2019) difieren de nuestros resultados ya que en su estudio implementaron una interfaz gráfica para probar los algoritmos de compresión más populares en los cuales destacan Huffman, Codificación aritmética, LZW y RLE. Sampson D. Asare y Mbewe Phyela (2019) rescatan en sus resultados que el algoritmo RLE tiene menos tiempo de compresión que el algoritmo LZW (0.001 y 0.002) respectivamente.

Por otro lado, para la tasa de compresión, los algoritmos IBK, LZW y RLE se obtuvieron los siguientes resultados: 0.5137, 0.3230 y 0.2526. Siendo algoritmo IBK el algoritmo que obtuvo mejor tasa de compresión, seguido del algoritmo LZW que obtuvo una mejor tasa de compresión que el algoritmo RLE.

Es asi que nuestro estudio concuerda con el estudio realizado por Sangeetha y Betty (2017) en donde se comparan seis algoritmos de compresión (RLE, Huffman, Delta Encoding, LZW, Transform Compression y MPEG) en imágenes biométricas, concluyendo que el algoritmo LZW tuvo una mejor tasa de compresión que los algoritmos descritos incluyendo al algoritmo RLE.

Por otro lado, Palanisamy et al. (2018) realizo un algoritmo hibrido entre LZW y Codificación aritmética para la compresión de imágenes multiespectrales y lo comparó con los algoritmos Huffman, RLE, LZW y codificación aritmética. Es así como dentro de sus resultados Palanisamy et al. (2018) obtuvieron que su algoritmo tuvo mejor resultado en tasa de compresión, pero cabe señalar que dentro de estos resultados el algoritmo LZW obtuvo mejores resultados que el algoritmo RLE ya que este obtuvo 0.8 comparado al 0.77 que obtuvo el algoritmo RLE.

Asi también nuestro estudio coincidió con el realizado por Birajdar Amit et al. (2019b) en su estudio previamente mencionado, obtuvieron que su optimización del algoritmo RLE fue mejor en imágenes en vertical que el algoritmo RLE, pero también Birajdar Amit et al. (2019b) añaden que el algoritmo LZW tuvo mejores resultados en tasa de compresión que el algoritmo RLE.

Nuestro estudio también concuerda con el realizado por Gupta y Nigam (2021) ya que en su estudio comparan a los algoritmos por el tipo de técnica empleada (basado en diccionarios y basados en entropía) y en base a sus

resultados Gupta y Nigam (2021) concluyen que el algoritmo LZW fue el que tuvo en mejor desempeño en el apartado de tasa de compresión.

Por otro lado, el estudio realizado por Beserra Antonio, Souza Leandro y Souza Daniel (2020) destacan que el algoritmo RLE se desempeñó mejor en archivos de texto y en archivos de video, mientras que el algoritmo LZW se desempeñó bien en todos los distintos tipos de archivos que se usaron en el estudio.

Sin embargo, nuestro estudio difiere del estudio llevado a cabo por Tianjiao Li et al. (2017) en el cual someten a los algoritmos RLE, LZW y a dos algoritmos híbridos entre LZW y RLE a la compresión de flujos de datos para observar en qué casos se desempeñan mejor. En los resultados obtenidos Tianjiao Li et al. (2017) en sus resultados obtienen que el algoritmo RLE tiene una mejor tasa de compresión que el algoritmo RLE, en donde el algoritmo RLE obtiene un 101% de tasa de compresión comparado al 57% del algoritmo LZW. Tianjiao Li et al. (2017) aclaran que los algoritmos RLE y LZW tienen sus ventajas y desventajas cuando son aplicados por separados ya que no se desenvuelven bien en todos los casos en la compresión de flujos de datos y que tambien sus algoritmos híbridos se desempeñaron mejor en ciertos casos mas no en todos.

Adedeji Kazeem (2020) en su estudio realizado, tambien concuerda con nuestro estudio en el apartado de tasa de compresión. Adedeji Kazeem (2020) compara los algoritmos RLE, LZW, Shannon-Fano y Huffman para la compresión de la data obtenida en una red de IoT de Smart Water, de esta manera en sus resultados obtiene que el algoritmo LZW fue mejor que los demás algoritmos con una compresión del 52% comparada con el 28% que obtuvo el algoritmo RLE en uno de los archivos de muestra.

Nuestro estudio concuerda con el estudio realizado por Punitha V. y Kalavathi P. (2020) ya que en su estudio realizan una comparación de los algoritmos RLE, LZW y Huffman en imágenes MRI. Punitha V. y Kalavathi P. (2020) obtienen en sus resultados que el algoritmo Huffman es mejor para comprimir imágenes MRI, seguido del algoritmo LZW, siendo el algoritmo RLE el que obtuvo un peor resultado.

Adicionalmente, en el apartado de tiempo de descompresión el algoritmo que obtuvo menor tiempo de descompresión fue el algoritmo LZW, seguido por el

algoritmo IBK y siendo el algoritmo RLE el algoritmo que obtuvo más tiempo de descompresión. Siendo los resultados obtenidos 0.02, 0.06 y 0.11 respectivamente.

Nuestro estudio difiere del estudio realizado por Rahman y Hamada (2019). En este estudio se comparan cinco algoritmos (RLE, Shannon-Fano, Huffman, LZW y Codificación aritmética) para la compresión de imágenes médicas creadas por computadora y otras obtenidas del DICOM data set. Rahman y Hamada (2019) obtuvieron como resultado de procesar las imágenes que el algoritmo RLE obtuvo un menor tiempo de descompresión con un tiempo de 0.059ms, comparado a los 0.009 del Algoritmo LZW.

El trabajo realizado por Adedeji Kazeem (2020) difiere con nuestro estudio ya que al procesar seis archivos de datos obtenidos de sensores Smart Water para el IoT, concluye que el algoritmo RLE fue el que tuvo mejor rendimiento que los otros algoritmos en cuestión de tiempo de descompresión ya que solo por nombrar el primer archivo, el algoritmo RLE obtuvo un tiempo de 110ms comparado a los 123ms conseguidos por el algoritmo LZW, los 150ms de Shannon-Fano y los 144ms del algoritmo Huffman.

VI. CONCLUSIONES

Las conclusiones de esta investigación fueron las siguientes:

1. El algoritmo IBK de compresión de imágenes obtuvo un promedio en la tasa de compresión de 0.5137 en la compresión de imágenes biométricas y obtuvo una mejor compresión que los algoritmos RLE, con 0.2526 y el algoritmo LZW, con 0.323.
2. El algoritmo IBK de compresión de imágenes obtuvo un promedio en el tiempo de compresión de 0.0923s en la compresión de imágenes biométricas, tiempo que fue superado por el algoritmo LZW con un tiempo de 0.900s pero que fue mejor que el tiempo de compresión del algoritmo RLE que obtuvo 0.1948s
3. El algoritmo IBK de compresión de imágenes obtuvo un promedio en el tiempo de descompresión de 0.0676s en la compresión de imágenes biométricas, tiempo que fue superado por el algoritmo LZW que obtuvo 0.0267s pero que fue mejor tiempo que el obtenido por el algoritmo RLE que obtuvo un tiempo de 0.1109s
4. En resumen, el algoritmo IBK obtuvo mejor radio de compresión que los algoritmos LZW y RLE. Sin embargo, no tuvo mejores tiempos de compresión ni de descompresión en general pero sí mejores tiempos de compresión y descompresión que el algoritmo RLE.

VII. RECOMENDACIONES

Las recomendaciones para las futuras investigaciones son las siguientes:

1. Aumentar el tamaño de la muestra. Ya que al aumentar el tamaño de las imágenes se podrían obtener unos resultados más precisos en todas las dimensiones medidas.
2. Utilizar data sets diferentes. Al usar otros data sets ayudaría a tener una mejor visión del desempeño de los algoritmos probados en este estudio.
3. Utilizar otro tipo de imágenes, podrían ser imágenes satelitales, imágenes MRI o tal vez imágenes DICOM
4. Hacer una comparación utilizando otros algoritmos de compresión, como podrían ser Shannon-Fano, Huffman, Codificación aritmética, etc.
5. Realizar una muestra que este compuesta de diferentes tipos de imágenes y que no esté constituida de un solo tipo de imágenes
6. Proponer una investigación en la que se comparen los algoritmos, pero en diferentes equipos con características distintas para comparar las similitudes y diferencias.
7. Realizar un estudio, pero utilizando imágenes a colores de diferentes tamaños y tipo.
8. Realizar un estudio usando imágenes con una resolución mayor.

REFERENCIAS

ADEDEJI KAZEEM, 2020. Performance Evaluation of Data Compression Algorithms for IoT-Based Smart Water Network Management Applications. *Journal of Applied Science & Process Engineering*, vol. 7, no. 2. ISSN 2289-7771.

AKINYEMI OMOLOLU, A. y MODINAT ABOLORE, M., 2019. Enhanced Image Compression Using the Lempel-Ziv-Welch Algorithm. . S.l.:

ALAM, M.A., AHSAN, F., SUBHANI, M., FAHMID HOSSAIN, F.M., ISLAM, M.S. y RUMA, K.N., 2018. Faster Image Compression Technique Based on LZW Algorithm Using GPU Parallel Processing. . S.l.: s.n., ISBN 9781538651636.

AL-JAWAHERRY, M.A. y HAMID, Y., 2021. Image Compression Techniques: Literature Review. *Journal of Al-Qadisiyah for Computer Science and Mathematics* [en línea], vol. 13, pp. 10. DOI 10.29304/jqcm.2021.13.4.860. Disponible en: https://doi.org/10.29304/jqcm.2021.13.4.860.

ALSHORMAN, O., 2020. MEDICAL IMAGE COMPRESSION TECHNIQUE FOR TELEMEDICINE APPLICATIONS. [en línea], vol. 15, no. 18. ISSN 1819-6608. Disponible en: www.arpnjournals.com.

ARORA, S. y KUMAR, G., 2018. Review of Image Compression Techniques. *International Journal of Recent Research Aspects*. S.l.:

BESERRA ANTONIO, SOUZA LEANDRO y SOUZA DANIEL, 2020. Bootstrap Analysis of Compression Algorithms. . S.l.:

BIRAJDAR AMIT, AGARWAL HARSH, BOLIA MANAN y GUPTE VEDANG, 2019a. Image Compression using Run Length Encoding and its Optimisation. . S.l.: s.n., ISBN 9781728136943.

BIRAJDAR AMIT, AGARWAL HARSH, BOLIA MANAN y GUPTE VEDANG, 2019b. Image Compression Using Run Length Encoding and Lempel Ziev Welch Method. ,

BOOPATHIRAJA, S., KALAVATHI, P. y DHANALAKSHMI, C., 2019. Significance of Image Compression and Its Upshots - A Survey. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pp. 1203-1208. DOI 10.32628/cseit1952321.

CAMILO SALAZAR, J., TOVAR, Á., CARLOS LINARES, J., LOZANO, A. y VALBUENA, L., 2018. Scrum contra XP: similitudes y diferencias. *TIA* [en línea], vol. 6, no. 2, pp. 29-37. ISSN 2344-8288. Disponible en: https://revistas.udistrital.edu.co/ojs/index.php/tia.

CHIANPHATTHANAKIT, C., BOONSONGSRIKUL, A. y SUPPHARANGSAN, S., 2018. A Lossless Image Compression Algorithm using Differential Subtraction Chain. . S.l.:

CRUZ DEL CASTILLO, C., OLIVARES OROZCO, S. y GONZÁLEZ GARCÍA, M., 2014. *Metodología de la investigación*. S.l.: s.n.

DAMIÁN CABEZAS MEJÍA, E. y ANDRADE NARANJO JOHANA TORRES SANTAMARÍA, D., 2018. *Introducción a la metodología de la investigación científica* [en línea]. S.l.: s.n. ISBN 978-9942-765-44-4. Disponible en: www.repositorio.espe.edu.ec.

FATHAHILLAH, S.G. y ZAIN, R., 2019. Homogeneous Image Compression Techniques with the Shannon-Fano Algorithm. *Int. J. Environ. Eng. Educ* [en línea], vol. 1, no. 2, pp. 59-66. DOI 10.5281/zenodo.3490205. Disponible en: https://doi.org/10.5281/zenodo.3490205.

FITRIYA, L.A., PURBOYO, T.W. y PRASASTI, A.L., 2017. A Review of Data Compression Techniques. *International Journal of Applied Engineering Research* [en línea]. S.l.: Disponible en: http://www.ripublication.com.

GONÇALVES LUIS, 2018. Scrum the methodology to become more agile. ,

GOPINATH, A. y RAVISANKAR, M., 2020. Comparison of Lossless Data Compression Techniques. *Proceedings of the 5th International Conference on Inventive Computation Technologies, ICICT 2020*. S.l.: Institute of Electrical and Electronics Engineers Inc., pp. 628-633. ISBN 9781728146850. DOI 10.1109/ICICT48043.2020.9112516.

GUPTA, A., BANSAL, A. y KHANDUJA, V., 2017. Modern lossless compression techniques: Review, comparison and analysis. *Proceedings of the 2017 2nd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2017*. S.l.: Institute of Electrical and Electronics Engineers Inc., ISBN 9781509032389. DOI 10.1109/ICECCT.2017.8117850.

GUPTA, A. y NIGAM, S., 2021. A Review on Different Types of Lossless Data Compression Techniques. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pp. 50-56. DOI 10.32628/cseit217113.

HERNÁNDEZ SAMPIERI, R., FERNÁNDEZ COLLADO, C. y BAPTISTA LUCIO, P., 2014. *Metodología de la investigación*. S.l.: s.n.

HRON, M. y OBWEGESER, N., 2018. *Scrum in practice: an overview of Scrum adaptations* [en línea]. S.l.: s.n. ISBN 9780998133119. Disponible en: http://hdl.handle.net/10125/50568.

KHANDWANI, F.I. y AJMIRE, P.E., 2018. A Survey of Lossless Image Compression Techniques. *International Journal of Electrical Electronics & Computer Science Engineering* [en línea], vol. 5, no. 1. ISSN 2454-1222. Disponible en: www.ijeecse.com.

LIU, X., AN, P., CHEN, Y. y HUANG, X., 2022. An improved lossless image compression algorithm based on Huffman coding. *Multimedia Tools and Applications*, vol. 81, no. 4, pp. 4781-4795. ISSN 15737721. DOI 10.1007/s11042-021-11017-5.

LOZANO, S., SUESCÚN, E., VALLEJO, P., MAZO, R. y CORREA, D., 2018. Comparando dos estrategias de aprendizaje activo para enseñar scrum en un curso introductorio de ingenieria de software. ,

MAHDI, M.S. y HASSAN, N.F., 2017. A Proposed Lossy Image Compression based on Multiplication Table. *Kurdistan Journal of Applied Research*, vol. 2, no. 3, pp. 98-102. ISSN 2411-7684. DOI 10.24017/science.2017.3.34.

MBEWE PHYELA y D. ASARE SAMPSON, 2017. Analysis and Comparison of Adaptive Huffman Coding and Arithmetic Coding algorithms. . S.l.: s.n., ISBN 9781538621653.

OLIVIA, L., VALDEZ, A., CRISPÍN, A., RUÍZ, G., LÓPEZ, E.J., LORENA, G., GUERRERO, D., CARLOS, J. y BRINDIS, V., 2014. Aplicación de la metodología semi-ágil ICONIX para el desarrollo de software: implementación y publicación de un sitio WEB para una empresa SPIN-OFF en el Sur de Sonora, México. . S.l.:

PALANISAMY, K., HIRAJA, B.S., KUMAR, A., BOOPATHIRAJA, S., KALAVATHI, P. y CHOKKALINGAM, S., 2018. A Hybrid Lossless Encoding Method for Compressing Multispectral Images using LZW and Arithmetic Coding. [en línea], ISSN 2347-2693. Disponible en: www.ijcseonline.org.

POOLAKKACHALIL THAFSEELA KOYA y CHANDRAN SARAVANAN, 2018. Analysis of Stereoscopic Image Compression using Arithmetic Coding and Huffman Coding. ,

PUNITHA V. y KALAVATHI P., 2020. Analysis of File Formats and Lossless Compression Techniques for Medical Images. [en línea], ISSN 2581-9283. Disponible en: https://www.researchgate.net/publication/342549114.

QASIM, A.J., DIN, R. y ALYOUSUF, F.Q.A., 2020. Review on techniques and file formats of image compression. *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 2, pp. 602-610. ISSN 23029285. DOI 10.11591/eei.v9i2.2085.

RAHMAN, M.A. y HAMADA, M., 2019. Lossless image compression techniques: A state-of-the-art survey. *Symmetry*, vol. 11, no. 10. ISSN 20738994. DOI 10.3390/sym11101274.

RAHMAN, M.A., HAMADA, M. y SHIN, J., 2021. The impact of state-of-the-art techniques for lossless still image compression. *Electronics (Switzerland)*, vol. 10, no. 3, pp. 1-40. ISSN 20799292. DOI 10.3390/electronics10030360.

ROSENBERG, D. y STEPHENS, M., 2007. Introduction to ICONIX Process. . S.l.: s.n.,

SAIDANI, A., XIANG, J. y MANSOURI, D., 2019. A New Lossless Compression Scheme for WSNs Using RLE Algorithm. . S.l.:

SALMAN, N.H. y HASSAN, E.Kh., 2019. Run Length Encoding Based Lossless MRI Image Compression Using LZW and Adaptive Variable Length Coding. *Journal of Southwest Jiaotong University*, vol. 54, no. 4. ISSN 0258-2724. DOI 10.35741/issn.0258-2724.54.4.23.

SAMPSON D. ASARE y MBEWE PHYELA, 2019. *CMedia Compressor: An Application to Graphically Compare General Compression Algorithms and Adaptive Huffman Compression Algorithm*. S.l.: IEEE. ISBN 9781538664773.

SANGEETHA, M. y BETTY, P., 2017. A Dynamic Image Compression using Improved LZW Encoding Algorithm. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology © 2017 IJSRCSEIT* [en línea], vol. 2, pp. 2017. ISSN 2456-3307. Disponible en: www.ijsrcseit.com.

SANGEETHA, M., BETTY, P. y NANDA, K., 2017. A Biometric Iris Image Compression using LZW and Hybrid LZW Coding Algorithm. . S.l.:

SARATHE, J., HAQUE, M.J., HUDA, M.N. y COORDINATOR, M., 2017. Study on Data Compression Technique General Terms Study on Data Compression Technique. *International Journal of Computer Applications*. S.l.:

SENTHILKUMARAN, N., THIMMIARAJA, J., VINODHINI, M., PROFESSOR, A. y SCHOLAR, R., 2017. A Study on Run Length Algorithm for Lossless Image Compression. . S.l.: s.n., ISBN 9788193331613.

SHARMA, N., 2018. A Study On Image Compression And Its Techniques. [en línea]. S.l.: Disponible en: http://iaetsdjaras.org/.

SURYA TEJA KODUKULLA, 2020. *Lossless Image compression using MATLAB* [en línea]. S.l.: s.n. Disponible en: www.bth.se.

TEODORO NICOMEDES, N.E., 2018. TIPOS DE INVESTIGACIÓN. . S.l.:

TIANJIAO LI, TIANDONG ZHAO, MINWOO NHO y XIAOFANG ZHOU, 2017. *A Novel RLE & LZW for Bit-Stream Compression*. S.l.: IEEE. ISBN 9781467397193.

YOUSIF, R.I. y SALMAN, N.H., 2021. Image compression based on arithmetic coding algorithm. *Iraqi Journal of Science*, vol. 62, no. 1, pp. 329-334. ISSN 23121637. DOI 10.24996/ijs.2021.62.1.31.

YUAN, S. y HU, J., 2019. Research on image compression technology based on Huffman coding. *Journal of Visual Communication and Image Representation*, vol. 59, pp. 33-38. ISSN 10959076. DOI 10.1016/j.jvcir.2018.12.043.

ANEXOS

Anexo 1: Matriz de Consistencia

PROBLEMAS	OBJETIVOS	HIPOTESIS	VARIABLES	DIMENSIONES	INDICADORES
General	General	General			
¿Qué algoritmo presenta la mejor tasa de compresión, tiempo de compresión y descompresión en imágenes entre los algoritmos IBK, LZW y RLE?	Determinar qué algoritmo presenta la mejor tasa de compresión, tiempo de compresión y descompresión en imágenes entre los algoritmos IBK, LZW y RLE.	El algoritmo IBK tendrá mejor tasa de compresión, tiempo de compresión y descodificación en imágenes comparado con LZW y RLE.	Efecto del algoritmo de compresión basado en codificación aritmética, LZW y RLE.		
Específicos	Específicos	Específicos		Compresión de imágenes. (Boopathiraja, Kalavathi y Chokkalingam 2018)	Tasa de compresión de imágenes. (Boopathiraja, Kalavathi y Chokkalingam 2018)
¿Qué algoritmo presenta la mejor tasa de compresión de imágenes entre los algoritmos IBK, LZW y RLE?	Determinar qué algoritmo presenta la mejor tasa de compresión de imágenes entre los algoritmos IBK, LZW y RLE.	El algoritmo IBK tendrá mejor tasa de compresión, en imágenes comparado con LZW y RLE. (Boopathiraja, Kalavathi y Chokkalingam 2018, Rahman y Hamada 2019)			
¿Qué algoritmo presenta el mejor tiempo de compresión de imágenes entre los algoritmos IBK, LZW y RLE?	Determinar qué algoritmo presenta el mejor tiempo de compresión de imágenes entre los algoritmos IBK, LZW y RLE.	El algoritmo IBK tendrá menor tiempo de compresión en imágenes comparado con LZW y RLE. (Rahman, Hamada y Shin 2021, Gupta, Khanduja y Bansal 2017)			Tiempo de compresión. (Rahman, Hamada y Shin 2021)
¿Qué algoritmo presenta el mejor tiempo de descompresión de imágenes entre los algoritmos IBK, LZW y RLE?	Determinar qué algoritmo presenta el mejor tiempo de descompresión de imágenes entre los algoritmos IBK, LZW y RLE.	El algoritmo IBK tendrá menor tiempo de dulcificación en imágenes comparado con LZW y RLE. (Rahman, Hamada y Shin 2021, Gupta, Khanduja y Bansal 2017)		Descompresión de imágenes. (Rahman, Hamada y Shin 2021)	Tiempo de descompresión. (Gupta, Khanduja y Bansal 2017)

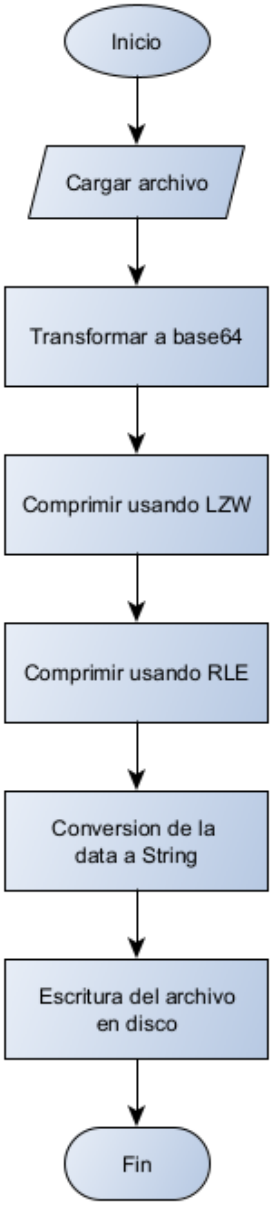
Tabla 20: Matriz de consistencia

Anexo 2: Matriz de operacionalización de variables

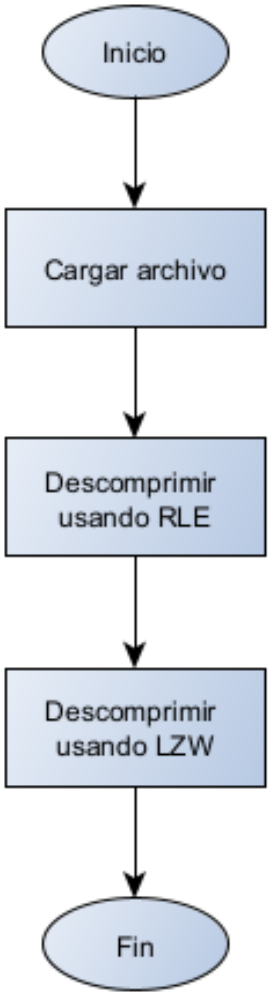
VARIABLES	DEFINICIÓN CONCEPTUAL	DEFINICIÓN OPERACIONAL	DIMENSIÓN	INDICADOR	INSTRUMENTO	ESCALA DE MEDICIÓN
Efecto del algoritmo de compresión basado en LZW y RLE.	La compresión de imágenes son un conjunto de técnicas que son aplicadas a las imágenes para después guardarlas de manera efectiva (2020, Qasim, Din and Alyousuf)	Técnicas de compresión de imágenes son mejores cuando contienen menos código en promedio, tiempo de compresión y descompresión y da un mejor radio de compresión (2019, Rahman and Hamada)	Compresión de imágenes. (Boopathiraja, Kalavathi y Chokkalingam 2019)	Tasa de compresión de imágenes. (Boopathiraja, Kalavathi y Chokkalingam 2019)	Ficha de Recolección de datos	Valor
				Tiempo de compresión. (Rahman, Hamada y Shin 2021)	Ficha de Recolección de datos	Valor
			Descompresión de imágenes. (Rahman, Hamada y Shin 2021)	Tiempo de descompresión. (Gupta, Khanduja y Bansal 2017)	Ficha de Recolección de datos	Valor

Tabla 21: Matriz de operacionalización de variables

Anexo 3: Diagrama de flujo del proceso de compresión del algoritmo IBK



Anexo 4: Diagrama de flujo del proceso de descompresión del algoritmo IBK



Anexo 5: Seudocódigo compresión de imágenes Algoritmo IBK

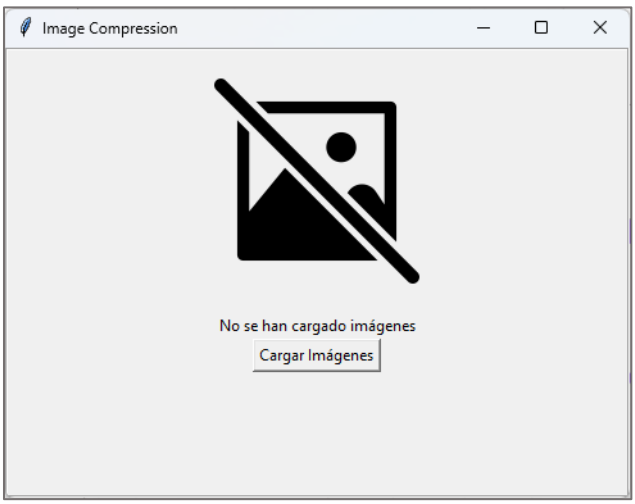
```
Inicio
  Leer archivo
  tabla = 2^256
  tamano_diccionario = 256
  diccionario = []
  para i = 0 ... i - 1 hacer:
    diccionario[i] = char(i)
  cadena = ''
  datos_comprimidos_LZW = []
  archivo = convertirBase64(archivo)
  para c en archivo:
    caracter = cadena + c
    Si caracter esta diccionario
      cadena = caracter
    Sino:
      insertar en datos_comprimidos_LZW(caracter)
      Si tamaño(diccionario) <= tabla
        diccionario[caracter] = tamano_diccionario
        tamano_diccionario = tamano_diccionario + 1
      cadena = c
  Si cadena esta diccionario
    insertar en datos_comprimidos_LZW(caracter)
  indice = 0
  RLE_compresion = []
  Mientras indice < tamaño(datos_comprimidos_LZW)
    contador = 1
    Mientras indice + 1 < tamaño(datos_comprimidos_LZW) &&
             datos_comprimidos_LZW[indice] == datos_comprimidos_LZW[indice + 1]
      contador = contador + 1
      indice = indice + 1
    insertar en RLE_compresion(String(datos_comprimidos_LZW[indice]+String(contador)))
    indice = indice + 1
  Escribir archivo con RLE_compresion con extension 'ibk'
```

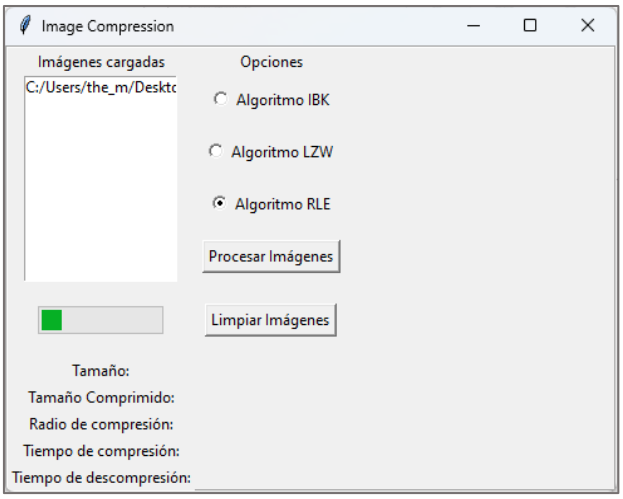
Anexo 6: Seudocódigo descompresión de imágenes Algoritmo IBK

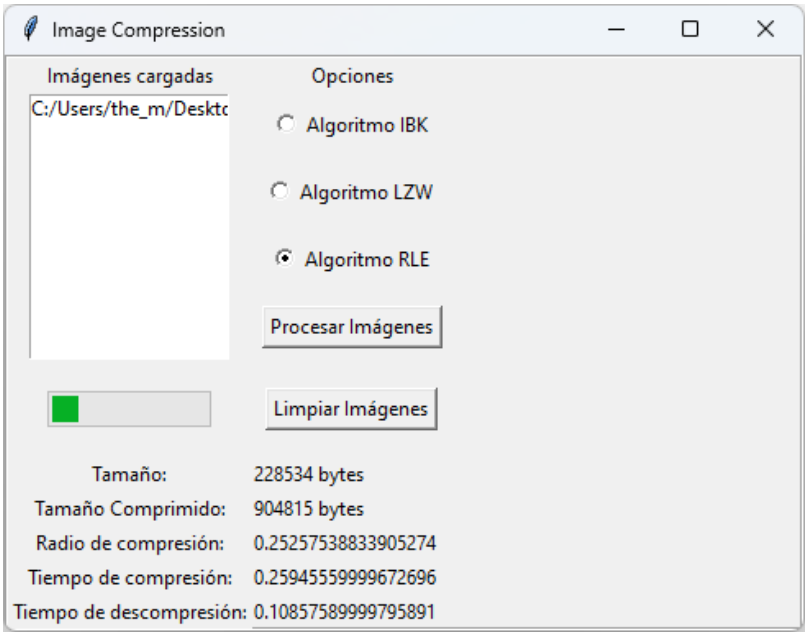
```
Inicio
  Leer archivo
  lzw_decode = []
  Para item en archivo:
    Para i en item:
      insertar en lzw_decode(item)
  ogData = ''
  tamano_diccionario = 256
  siguiente = 256
  diccionario = diccionario(character en tamano_diccionario)
  cadena = ''

  Para l en lzw_decode:
    Si no l en diccionario:
      diccionario[l] = cadena + cadena[0]
    insertar en lzw_decode(lzw_decode + diccionario[l])
    Si no tamaño(cadena) == 0
      diccionario[siguiente] = cadena + diccionario[l][0]
      siguiente = siguiente + 1
    cadena = diccionario[l]
Fin
```

Anexo 7: Prototipos







Anexo 8: Instrumento de recolección de datos

N.º	NOMBRE DE ARCHIVO	TAMAÑO	TAMAÑO COMPRIMIDO	RADIO DE COMPRESION	TIEMPO DE COMPRESION	TIEMPO DE DESCOMPRESION

Anexo 9: Metodología SCRUM

Hron y Obwegeser (2018) menciona que Scrum es uno de los frameworks más populares del desarrollo ágil debido a su simplicidad y versatilidad. Gonçalves Luis (2018) menciona que Scrum es un framework que es usado principalmente para desarrollo de productos y de software y que además es perfecto para entornos complejos en donde los equipos tienen que reaccionar y adaptarse de manera rápida a nuevas situaciones.

Camilo Salazar et al. (2018) señalan que Scrum es una metodología que es iterativa ya que se ejecuta en intervalos cortos y fijos e incremental debido a que se añaden más funcionalidades al terminar estos intervalos. Gonçalves Luis (2018) señala que Scrum está constituido por unas iteraciones llamadas sprints y que estos a su vez pueden contener diferentes eventos. Estos sprints son llevadas a cabo por un equipo de Scrum que pueden tener diferentes roles como lo son el Product Owner, Scrum Master y el equipo de desarrollo. Gonçalves Luis (2018) indica que un sprint es un periodo de tiempo de máximo un mes en donde se realiza un producto potencialmente entregable y que además un sprint solo puede iniciar si el anterior ha terminado.

- Product Owner: Gonçalves Luis (2018) define al Product Owner como el responsable de maximizar el trabajo del equipo de desarrollo para entregar un producto de calidad. A su vez de administrar el Product Backlog.
- Scrum Master: Camilo Salazar et al. (2018) define al Scrum Master como aquel integrante que se encarga de administrar los procesos y garantizar que el equipo cuente con todo lo necesario para realizar su función.
- Development Team: Gonçalves Luis (2018) lo define como un conjunto de profesionales que trabajan para entregar un producto de alta calidad al finalizar cada Sprint.

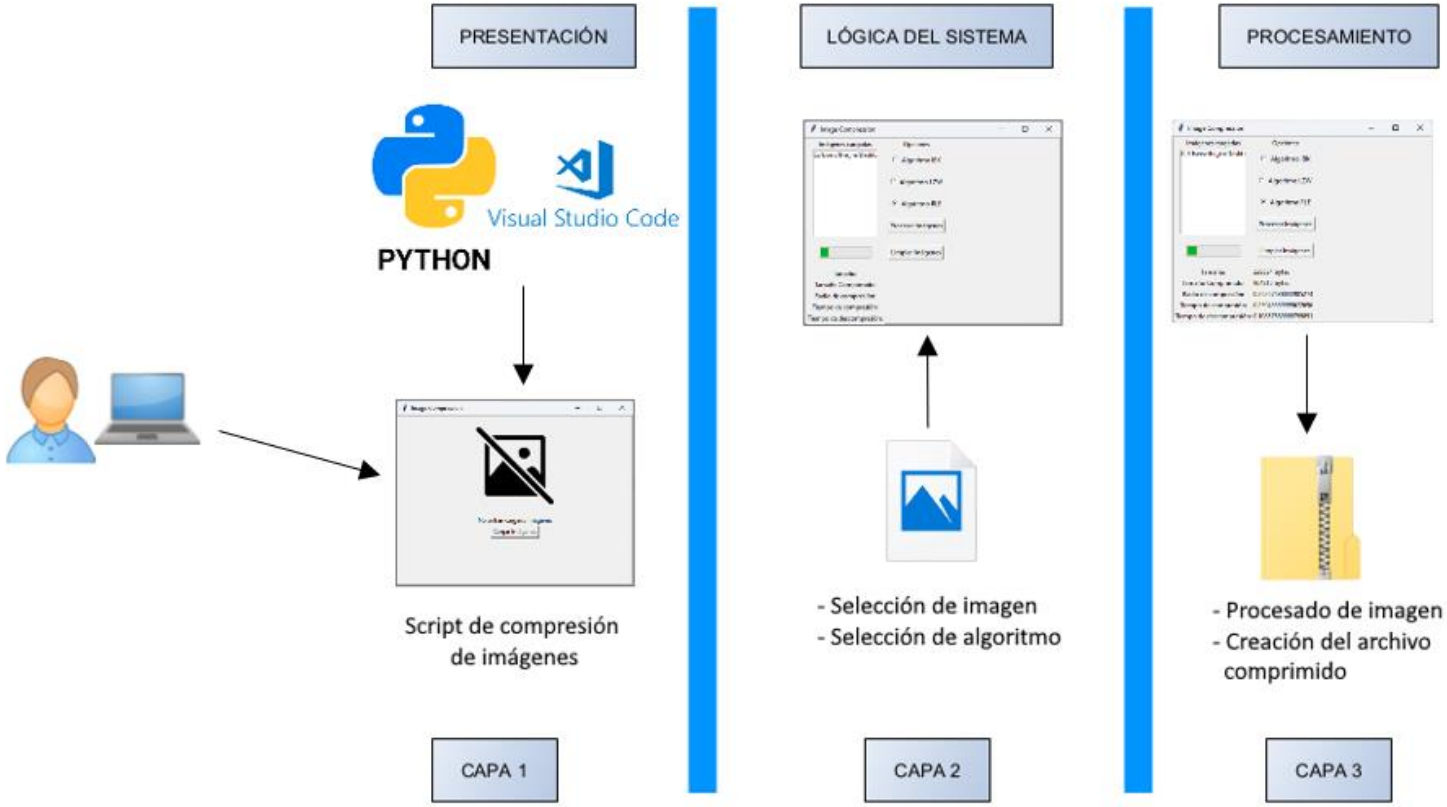
Adicionalmente a esto Camilo Salazar et al. (2018) que en Scrum hay una serie de artefactos que permiten llevar un registro del proyecto a través del tiempo, que vendrían a reemplazar formatos que se sigan al pie de la letra.

- Product Backlog: Gonçalves Luis (2018) señala que el Product Backlog es una lista con todos los requerimientos que el producto debe incluir al final. También debe incluir cualquier corrección que sea necesaria.
- Sprint Backlog: Camilo Salazar et al. (2018) mencionan que el Sprint Backlog contiene todas las tareas necesarias para realizar un ítem del Product Backlog, además debe contener un estimado en tiempo y el encargado de realizarla.
- Increment: Gonçalves Luis (2018) indica que esto es el resultado de un Sprint Backlog y que representa un paso adelante en el proceso de desarrollo
- Burndown Chart: Camilo Salazar et al. (2018) señalan que este artefacto permite apreciar el estado actual del proyecto.

Por último, Gonçalves Luis (2018) indica que los Scrum Events son eventos formales y que todos deben tener una duración máxima fija.

- Sprint Planning: Gonçalves Luis (2018) menciona que el Sprint Planning involucra a todos los miembros del equipo y que el equipo de desarrollo señala que es lo que realmente es realizable durante el Sprint. También se discute el objetivo del Sprint.
- Daily Scrum: Gonçalves Luis (2018) indica que el Daily Scrum se lleva a cabo en sesiones de 15 minutos para sincronizar las actividades del día.
- Sprint Review: Gonçalves Luis (2018) menciona que una vez que el Sprint ha sido completado, esta reunión se tiene que llevar a cabo. En ella se discuten los logros del Sprint y que ha quedado pendiente desde los otros Sprints.
- Sprint Retrospective: Gonçalves Luis (2018) menciona que es una reunión que es llevada a cabo después de un Sprint Review y antes de una planeación de un Sprint nuevo. En esta reunión se da la oportunidad de inspeccionar lo que está pendiente y que se podría llevar a cabo en el próximo Sprint.

Anexo 10: Arquitectura tecnológica



UNIVERSIDAD CÉSAR VALLEJO

FACULTAD DE INGENIERÍA Y ARQUITECTURA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

Declaratoria de Autenticidad del Asesor

Yo, CHUMPE AGESTO JUAN BRUES LEE, docente de la FACULTAD DE INGENIERÍA Y ARQUITECTURA de la escuela profesional de INGENIERÍA DE SISTEMAS de la UNIVERSIDAD CÉSAR VALLEJO SAC - LIMA ESTE, asesor de Tesis titulada: "Algoritmo para la compresión de imágenes basado en algoritmo LZW y RLE", cuyo autor es BORJA CASTAÑEDA KEVIN ARNOLD, constato que la investigación tiene un índice de similitud de 22.00%, verificable en el reporte de originalidad del programa Turnitin, el cual ha sido realizado sin filtros, ni exclusiones.

He revisado dicho reporte y concluyo que cada una de las coincidencias detectadas no constituyen plagio. A mi leal saber y entender la Tesis cumple con todas las normas para el uso de citas y referencias establecidas por la Universidad César Vallejo.

En tal sentido, asumo la responsabilidad que corresponda ante cualquier falsedad, ocultamiento u omisión tanto de los documentos como de información aportada, por lo cual me someto a lo dispuesto en las normas académicas vigentes de la Universidad César Vallejo.

LIMA, 15 de Diciembre del 2022

Apellidos y Nombres del Asesor:	**Firma**
CHUMPE AGESTO JUAN BRUES LEE **DNI:** 44824114 **ORCID:** 0000-0001-7466-9872	Firmado electrónicamente por: JCHUMPEA el 16-12-2022 22:34:21

Código documento Trilce: TRI - 0490483

INVESTIGA UCV